VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

2017 Michal Přikryl

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Michal Přikryl**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: Vikipid a.s.
2. Struktura závěrečné zprávy:
   a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
   b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
   c) Zvolený postup řešení zadaných úkolů.
   d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
   e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
   f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Daniel Stříbný**

Konzultant bakalářské práce: Bc. Michal Soviak

Datum zadání: 01.09.2016

Datum odevzdání: 28.04.2017

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 25. dubna 2017 .....

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.

V Ostravě 25. dubna 2017

Rád bych poděkoval Ing. Danielovi Střibnému za odbornou pomoc při tvorbě této bakalářské práce, jednateli firmy VIKIPID a.s. Davidu Chudějovi za umožnění odborné praxe a Bc. Michalovi Sloviakovi, který byl mým konzultantem na odborné praxi, za předané zkušenosti.

Abstrakt

Tato bakalářská práce popisuje průběh mé pracovní činnosti u společnosti VIKIPID a.s., kde jsem vykonával svou odbornou praxi. V úvodu popisuji, proč jsem si vybral tuto práci a společnost. V druhé části stručně popisuji společnost VIKIPID, a také svou pracovní pozici. V další části popisuji úkoly, na kterých jsem pracoval. Ve čtvrté části detailněji popíši mnou zvolené řešení jednotlivých úkolů. V závěru zhodnotím celkový přínos praxe, získané dovednosti a zkušenosti, a zároveň i chybějící znalosti.

Klíčová slova: bakalářská práce, odborná praxe, VIKIPID a.s., C#, .NET Framework, MVC, HTML

Abstract

This bachelor thesis describes the course of my work activities at the company VIKIPID a.s., where I did my internship. In the beginning, I am going to describe why I chose this intern position and this particular company. In the second section, I'm going to describe the company in question, namely VIKIPID, and my own position. Furthermore, I am going to explain the tasks which I have been working on. In the fourth section, I am going to describe in detail my chosen solution of individual tasks. Last but not least, I am going to evaluate and reflect on the overall benefit of the internship, the experience I have gained, and the lack of knowledge.

Key Words: bachelor thesis, professional practice, VIKIPID a.s., C#, .NET Framework, MVC, HTML

Obsah

Seznam použitých zkratek a symbolů

ASP.NET	–	Active Server Pages .NET
MVC	–	Model View Controller
HTML	–	Hyper Text Markup Language
HTTP	–	Hypertext Transfer Protocol
HTTPS	–	Hypertext Transfer Protocol Secure
XML	–	Extensible Markup Language
JSON	–	JavaScript Object Notation
SOAP	–	Simple Object Access Protocol
WCF	–	Windows Communication Foundation
REST API	–	Representational State Transfer Application Programming Interface
WebAPI	–	Web Application Programming Interface
WSDL	–	Web Description Language
IIS	–	Internet Information Services
TFS	–	Team Foundation Server
SŘBD	–	Systém řízení báze dat
SQL	–	Structured Query Language

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Při výběru tématu bakalářské práce mě zaujala možnost absolvování odborné praxe ve firmě. Po rozvaze jsem si proto jako téma své bakalářské práce zvolil tuto možnost. Nejvíce se mi zamlouvala možnost získat touto formou praxi ve svém oboru, která je velmi důležitá k uplatnění po ukončení studia. Praxe mi umožnila rozvíjet mé znalosti do hloubky a v neposlední řadě získat praktické zkušeností z oboru. Dalším důvodem byl i můj zájem o webové technologie ASP.NET (respektive MVC) i .NET Framework obecně.

Svou odbornou praxi jsem od srpna 2016 vykonával ve společnosti VIKIPID a.s. V první části této práce popíši společnost VIKIPID a mou roli v této společnosti. V druhé části se budu zabývat popisem jednotlivých úkolů, které jsem dostal přiděleny. U všech úkolů vždy nastíním problematiku týkající se zadání. Ve třetí části rozeberu mnou zvolené řešení a také detailněji osvětlím problematiku zadaných úkolů. V závěru této práce zhodnotím využití mých znalostí nabytých při studiu a znalostí, které jsem získal praxí. Na samotném konci zhodnotím celkový přínos odborné praxe.

2 Popis odborného zaměření firmy a pracovního zaměření

2.1 Profil firmy

Společnost VIKIPID a.s.[1] vznikla v Ostravě ke konci roku 2012. Jejím záměrem je uvést na český trh portfolio služeb, které zvýší bezpečnost nakupování na internetu, a zároveň obchodníkům zlevní a zefektivní obchodování v tomto podnikatelském prostředí.

Produktem firmy je systém VIKIPID, který zaštiťuje soubor služeb potřebných pro efektivní obchodování na internetu. Systém VIKIPID je vytvořen v jazyce C# na platformě .NET, webové aplikace jsou postaveny na technologii ASP.NET MVC. Systém je neustále rozvíjen o nové funkce a možnosti, jak pro zákazníky, tak pro uživatele.

Obrázek 1: Logo společnosti VIKIPID a.s.

2.2 Pracovní pozice studenta

Ve firmě jsem pracoval jako .NET programátor. Pracoval jsem v týmu 3 osob. Mým hlavním úkolem bylo vytvoření webových rozhrání, rozhrání pro komunikaci s bankovními systémy, tvorba dokumentace a v neposlední řadě také úpravy struktur SQL Server databáze. Součástí mé práce bylo i přetváření úkolů vedení společnosti v oblasti funkčnosti systému na reálné části systému, to znamená výběr vhodné technologie a zpracování.

3 Seznam zadaných úkolů

V této kapitole jsou vypsány jednotlivé úlohy, kterými jsem se v průběhu odborné praxe zabýval. Úlohy jsou seřazeny dle pořadí, v jakém byly zadávány.

3.1 Napojení na systém Multicash

Úkolem bylo navrhnout a implementovat propojení mezi systémem VIKIPID a bankovním systémem Multicash. Propojení spočívá ve vytváření a přijímání datových souborů, které se dále zpracovávají programem Multicash. Součástí bylo také vytvoření vhodných datových struktur v databázi.

3.2 Napojení na systém dopravce INTIME

Cílem bylo zajistit automatické odesílání informací, o uživatelem vybraných zásilkách, do Aplikace zásilky přepravní společnosti INTIME. Komunikace probíhá pomocí HTTP protokolu, data jsou zasílány ve formátu XML i JSON. Součástí bylo taktéž automatizované objednávání svozu, po předchozí žádosti uživatele. Dále také zjišťování aktuálního stavu zásilky, která již byla převzata dopravcem a je na cestě k zákazníkovi e-shopu (Track & Trace).

3.3 Vstupní aplikace pro příjem objednávek ze systému e-shopu

V systému VIKIPID se pracuje s objednávkami zákazníků e-shopu. Informace o těchto objednávkách se budou v systému přijímat pomocí webových aplikací. Mým úkolem bylo dle požadavků vytvořit tyto webové aplikace tak, aby nevyžadovaly složitou implementaci ze strany e-shopů a zároveň vyhověly co nejvíce požadavkům na datové formáty.

3.4 Implementace eAPI platební brány ČSOB

Mým dalším úkolem bylo propojit systém VIKIPID a eAPI ČSOB. Díky tomuto propojení je možné v systému VIKIPID platit za objednávky pohodlně pomocí platební karty. Komunikace probíhá pomocí HTTPS protokolu. Součástí tohoto úkolu, bylo i vytvoření webové aplikace VIKIPID Brána, pomocí které zákazník e-shopu přistupuje k platební bráně ČSOB a jejím službám.

3.5 VIKIPID Intranet

Systém VIKIPID je spravován zaměstnanci skrze webovou aplikaci VIKIPID Intranet. Mým úkolem bylo vytvořit tuto aplikace se vzhledem, podobným VIKIPID Účtu. Součástí úkolu bylo také implementování většiny funkcí, které usnadňují zaměstnancům správu systému prostřednictvím této webové aplikace.

3.6 VIKIPID Účet

VIKIPID Účet je prostředí pro zákazníky VIKIPIDu (obchodníky). Zde mohou spravovat své objednávky a zjistit další užitečné informace. Mým úkolem bylo vytvořit části týkající se svozových míst, správy uživatelů, agendu smluvních dokumentů a hlavní nástěnku. Součástí tohoto úkolu bylo také vytvoření uživatelské dokumentace, která je určena primárně zákazníkům - obchodníkům.

3.7 Časová náročnost jednotlivých úkolů

Úloha	Počet dní
Napojení na systém Multicash	14
Napojení na systém dopravce INTIME	10
Vstupní aplikace pro příjem objednávek	6
Implementace eAPI platební brány ČSOB	20
VIKIPID Intranet	25
VIKIPID Účet	14

Tabulka 1: Časová náročnost jednotlivých úkolů

4 Postup řešení zadaných úkolů

4.1 Napojení na systém Multicash

Můj první úkol spočíval ve vytvoření propojení mezi systémem VIKIPID a systémem Multicash. Systém, respektive program, Multicash zprostředkovává propojení mezi informačními systémy bankovních zákazníků a systémem banky, jedná se v podstatě o takové zjednodušené internetové bankovnictví. Lze v něm zadávat různé platby, jak české, tak zahraniční. Důležitou vlastností je přijímání informací o pohybech na účtu, a to dokonce i dříve, než je platba (částka) fyzicky přítomna na účtu - připočítána k zůstatku. Časová náročnost tohoto úkolu byla 14 dní, převážně z důvodu nastudování formátu souborů a obecného testování napojení.

4.1.1 Základní implementace

Výměna dat mezi informačními systémy zákazníků bank a systémem Multicash probíhá na základě souborů. Soubory musí být v textovém formátu s kódováním CP852. Soubory se dále liší pouze ve formátu dat pro jednotlivé druhy plateb. Jako první jsem vytvořil návrh databázové struktury, který pokryl požadavky zadání na uchovávaní historických dat. Na základě databázové struktury jsem vytvořil třídní diagram tak, aby vhodně obsáhl, jak datové požadavky, tak i co nejvíce generické vytváření jednotlivých formátů platebních souborů.

Rozdělení formátu souborů

- CFD - formát pro tuzemské platby v CZK
- CFU - formát pro urgentní tuzemské platby v CZK
- CFA - formát pro zahraniční platby v jiné měně než CZK a EUR
- CCT - formát pro zahraniční platby v EUR (SEPA platby)
- MT900 - formát pro informování o bankou zamítnuté inkasní platbě
- MT940 - formát pro stavy účtu
- MT942 - formát pro avíza o pohybech na účtu

Druhou částí návrhu bylo zpracování příchozích souborů ze systému Multicash. Tyto soubory obsahují informace o přijatých a odchozích platbách v rámci jednotlivých bankovních účtů. Dříve než je, jak příchozí, tak i odchozí platba, zohledněna v zůstatku účtu, přijme program Multicash z banky avízo o této platbě. Avízo znamená informaci, že v horizontu 1-2 dní přijde/odejde avízem definovaná platba na určitý účet. Všechny tyto údaje je také nutné vhodně ukládat, jak pro historickou kontrolu, tak pro automatickou správu bankovních účtů. Databázové struktury pro stavy účtu a avíza jsou jednotné pro všechny bankovní ústavy. První problém nastal při

vytváření třídních diagramů, kdy jsem z dokumentací jednotlivých bank vyčetl více či méně výrazné rozdíly ve formátu dat v souborech, tudíž nebylo možné zvolit generické zpracování, nýbrž se soubory musí zpracovávat odděleně pro každou banku.

4.1.2 Implementace odesílání plateb

Hlavním úkolem bylo vytvořit generátor plateb, který vytváří platby na základě dat zadaných do databáze servisem, starajícím se o finanční operace v systému VIKIPID. Tyto platby mohou probíhat v různých měnách, dle této měny je potřeba zvolit správný datový formát, který je nutné bezpodmínečně dodržet, jinak banka odmítne tuto platbu přijmout.

Jak jsem již zmínil, formát souboru je textový, kódování je CP852 a jsou povoleny pouze velké písmena bez diakritiky. Formáty využívané pro platby jsou CFD, CFU, CFA a CCF. Pro generování souborů (zápis do souboru) jsem zvolil standardní třídu .NET Frameworku System.IO.StreamWriter. Před samotným zápisem je potřeba upravit data načtená z databáze do tvaru vhodného pro vybraný formát. Tyto úpravy zahrnují například odstranění diakritiky pomocí knihovny System.Globalization a zkrácení délky názvů (bankovního účtu aj.). Důležité je také hlídat správný formát čísla bankovního účtu, které je buď v nejčastěji užívaném číselném formátu nebo ve formátu IBAN, pro oba druhy zápisy čísla bankovního účtu existují kontrolní algoritmy, které jsou aplikovány před vytvářením platby a samotným zápisem do souboru. V případě, že některé z dat určených k zápisu nejsou shledány jako validní, je tato platba vyřazena a pomocí chybových zápisů je vývojář upozorněn na vzniklou situaci, na kterou musí co nejdříve reagovat, jinak by došlo například k nechtěnému prodlení při platbě. Po úspěšném vytvoření souboru s platbou/ami se soubor automaticky přesune do předem vytyčené složky, kde jej již zpracuje program Multicash.

```
private int Mod(string iban, int from = 0, int count = 9, string remain = "")
{
   int res = int.Parse($"{remain}{iban.Substring(from, count)}") % 97;
   if (iban.Length == from + count)
   {
      return res;
   }
   else
   {
      return Mod(iban, from + count, Math.Min(iban.Length - from - count,
         res > 9 ? 7 : 8), res.ToString());
   }
}
```

Výpis 1: Rekurzivní metoda pro výpočet kontrolního součtu IBAN

4.1.3 Implementace přijímání a zpracování souborů

Druhou částí úkolu bylo zpracování stavů účtů a avíz o příchozích a odchozích platbách. Komunikace při těchto úkonech probíhá taktéž na základě souborů. Program Multicash tyto soubory ukládá do předem určené složky, odkud jsou po zpracování automaticky přesunuty do složky se zpracovanými soubory. Tyto soubory jsou v textovém formátu, kódovány pomocí CP852. Bohužel struktura těchto souboru je více či méně rozdílná pro každou banku - každá banka má některé pole uzpůsobené pro své potřeby - a tuto skutečnost bylo důležité zohlednit při vytváření třídního návrhu. Další komplikací bylo, že ne vždy bankovní dokumentace odpovídala skutečnému formátu souborů, tudíž bylo nutné vyčíst reálný formát z příchozích souborů.

Soubor je načítán po řádcích pomocí třídy .NET Frameworku System.IO.StreamWriter a posléze jsou získávány požadované informace parsováním jednotlivých řádků. Každá banka je představována třídou, ve které se data zpracují a poté ukládají do databáze. Stavy účtů i avíza jsou děleny podle bankovního účtu s nímž jsou spjaty, kdy například s pomocí získaných údajů lze vypočíst aktuální i předpokládaný finanční zůstatek na účtu, aby bylo možné automaticky udržovat dostatečný zůstatek pro manipulaci s finančními prostředky. Důležitou vlastností avíz je získání informace o budoucím příchodu platby za objednávku (v případě platby bankovním převodem), kdy jsou platba a objednávka spárovány pomocí variabilního symbolu, čímž se docílí dřívějšího získání informace o zaplacení objednávky, než je například možné zjistit z internetového bankovnictví.

```
CEKOCZPPAXXX 00000
942 01

:20:VIKIPID a.s.
:25:0600/1817450567
:34F:PLND0,
:61:170221C120,00NMSC
:86:111?20ZUSTATEK Z 21.2.2017 07:01

CEKOCZPPAXXX 00000
942 01
```

Výpis 2: Ukázka formátu souboru avíza (MT942)

4.2 Napojení na systém dopravce INTIME

Dalším úkolem bylo implementovat automatickou správu zásilek, které jsou přiřazeny dopravci INTIME. Pod automatickou správou se skrývá odeslání dat o zásilce do systému dopravce, objednání svozu na vybrané zásilky a den, a sledování stavu zásilky. Vedlejším úkolem byla automatická aktualizace databáze adres Poštomatů, kterou je důležité udržovat denně aktuální, jelikož seznam Poštomatů se může každý den měnit.

4.2.1 Import dat do Aplikace zásilky

Prvním krokem při zpracování zásilky je odeslání dat do Aplikace zásilky (dále jen AZ). Implementace importu dat trvala přibližně okolo 7 dní. Komunikace je zde založena na protokolu HTTP a jeho metodě POST. Data jsou předávána ve formátu XML. Zde jsem se rozhodoval, zda XML vytvářet po jednotlivých elementech nebo zda jej vytvořit serializací objektu, který by odpovídal svou strukturou požadované struktuře dat v AZ. Bohužel formát dat, ve kterém je možné zasílat proměnný počet zásilek v jednom požadavku, znemožnil snadné užití serializace, tudíž jsem přistoupil k vytváření XML po jednotlivých elementech, pro vytváření jsem využil třídy z jmenného prostoru System.Xml, které poskytují vhodné třídy a metody pro práci s XML. Vytváření požadavku po jednotlivých elementech na mě působilo i přehledněji, jelikož jsem měl možnost upravit formát každého jednotlivého elementu.

Po načtení dat o zásilce z databáze a zpracování dat do XML přichází na řadu odeslání dat pomocí třídy System.Net.HttpWebRequest, která poskytuje metody pro práci s protokolem HTTP. Po odeslání dat je nutné zkontrolovat přijetí dat do AZ. Pokud byla zásilka v AZ úspěšně přijata, je objednávce přiřazen přepravní kód. Pokud je zásilka AZ odmítnuta, je programátor upozorněn v chybovém logu systému, a navíc je objednávka označena jako chybová.

Dalším krokem v životním cyklu objednávky je fyzické předání zásilky dopravci. Tomu předchází objednání svozu na určitý den a čas. Tento den a čas si předem zvolí obchodník ve svém VIKIPID Účtu, až poté je svoz automaticky objednán. Mým úkolem bylo implementovat automatické odesílání této žádosti o svoz. Komunikace probíhá taktéž, jako u importu dat o zásilce, na základě protokolu HTTP a metody POST, a data jsou také zasílána v XML. V podstatě jde o podobný úkon, jako u importu dat o zásilce, jediný rozdíl je ve formátu odesílaného XML.

4.2.2 Track & Trace INTIME

Posledním krokem byla implementace získávání stavů zásilek z aplikace Track & Trace. Tento úkol nebyl zdaleka tak složitý, jako jiné úkoly, naopak byl spíše rozsáhlý, z důvodu implementace dvou služeb. Kompletní implementace obou služeb trvala 3 dny. Společnost INTIME nabízí dva způsoby získávání stavů zásilky. Oba způsoby jsou založeny na protokolu HTTP a jeho metodě GET, kdy je odeslán GET požadavek a v návaznosti na něj je přijata odpověď. K odeslání požadavku jsem využil novější třídy .NET Frameworku System.Net.Http.HttpClient, kterou

je dnes nahrazována starší třída System.Net.HttpWebRequest. Požadavek je vracen v jednom případě ve formátu XML a v druhém případě ve formátu JSON.

U formátu JSON jsem přistoupil k deserializaci do objektu, který jsem si vytvořil dle dokumentace, tuto deserializaci zprostředkovává třída Newtonsoft.Json. Program poté rozklíčuje dané stavy objednávky a pokud se objevil nějaký nový stav, uloží jej do databáze a případně změní stav objednávky. U formátu XML jsem byl nucen znovu místo deserializace zvolit postupné rozklíčování odpovědi, jelikož formát XML nebyl vhodný pro deserializaci. Pro jednodušší vyhledání v XML jsem využil XPath, který slouží pro adresaci XML dokumentu, vybírání jednotlivých elementů a práci s hodnotami a atributy těchto elementů[2]. Poté program taktéž rozklíčuje stavy objednávek, uloží je do databáze a případně změní stav objednávky. V případě přijetí některých stavů, je také zákazník e-shopu (příjemce objednávky) informován e-mailem o změně stavu zásilky. Všechny dříve zmiňované stavy objednávek si má obchodník možnost zobrazit ve svém VIKIPID Účtu.

```
string url = _trackAndTraceURL.Replace("{number}", order.TransportCode);
using (HttpClient client = new HttpClient())
{
    var task = client.GetAsync(url).ContinueWith(response =>
    {
        var result = response.Result;
        var xmlstring = result.Content.ReadAsStringAsync();
        xmlstring.Wait();
        res = xmlstring.Result;
    });
    task.Wait();
}
```

Výpis 3: Asynchronní odeslání GET požadavku na službu INTIME Track & Trace

4.3 Vstupní aplikace pro příjem objednávek ze systému e-shopu

Jednou z hlavních podstat systému VIKIPID je administrace objednávek, kdy tyto objednávky musí být do systému importovány. Mým dalším úkolem tedy bylo vytvořit webovou aplikaci pro příjem objednávek přímo z informačních systémů e-shopů. Součástí tohoto úkolu bylo i vytvoření dokumentace, popisující struktury využitých objektů, které reprezentují žádost i odpověď při komunikaci se službou, spolu se seznamem datových omezení jednotlivých vlastností objektu.

4.3.1 Základní implementace

Jako první jsem musel zvolit, s pomocí jaké technologie bude webová aplikace komunikovat. Zvolil jsem dnes preferované REST API (ASP.NET MVC WebAPI) a SOAP komunikaci (Web Service - WCF). WebAPI jsem zvolil z důvodu, že umožňuje vytvořit RESTfulAPI, které podporuje standardní metody protokolu HTTP. Web Services jsem zvolil z toho důvodu, že podporuje mírně odlišný způsob komunikace založený na protokolu SOAP (i když také pomocí protokolu HTTP) a nabízí popis rozhraní pomocí jazyka WSDL. WebAPI akceptuje objekty ve formátu XML nebo JSON, zatímco Web Service akceptuje objekty pouze ve formátu XML (SOAP). Data se odesílají do obou aplikací pomocí metody POST.

Posléze bylo nutné vytvořit seznam, který obsahuje výčty hodnot pro jednotlivá pole, povolené kombinace hodnot polí a také povinnost vyplnění polí. Na základě tohoto seznamu jsem vytvořil strukturu objektu, který bude odesílán z e-shopů na rozhraní výše zmíněných služeb. V jednom takovém objektu je možné zaslat libovolný počet objednávek. Přístup do služby je zabezpečen uživatelským jménem a heslem, které je přiřazeno e-shopu při registraci do systému VIKIPID.

4.3.2 Implementace společné třídy obou rozhraní

Obě služby využívají společnou třídu, ve které se zpracovává objekt objednávky a poté, v případě, že jsou jeho hodnoty správné, se objednávka ukládá do databáze. Služby mají návratový objekt, ve kterém se nachází veškeré zaslané objednávky, a navíc je ke každé objednávce, v případě špatného formátu, či jiné chyby, přidána kolekce obsahující objekty s názvem a stručným popisem chyby, aby byla možná zpětná kontrola na straně e-shopu. Kontrola vstupního objektu objednávky je koncipována tak, aby zachytila maximální možný počet formátových chyb (v případě výskytu chyb) ještě před samotným odesláním odpovědi zpět do systému e-shopu. Tato kontrola je zavedena z důvodu zachycení co největšího množství chyb, již při prvotním odeslání požadavku, aby se eliminoval počet zaslaných chybných požadavků.

Taktéž celý návratový objekt obsahuje navíc zprávu o úspěšném přijetí požadavku, případně popis chyby, která nastala při zpracování (například nesprávný formát požadavku, nesprávné přihlašovací údaje, chybějící povinná data aj.).

4.3.3 Nasazení a testování aplikací

Součástí úkolu bylo i nasazení aplikací na webový server (IIS), vytvoření poddomény pod vikipid.cz a otestování funkčnosti aplikací. Pro testování funkčnosti jsem si vytvořil třídu, která vytvoří objekt objednávky a poté jej odešle, buď ve formátu JSON nebo XML, pomocí metody POST, na rozhraní webové aplikace. Při testování odesílání požadavku ve formátu XML jsem zjistil, že WebAPI chybně deserializuje zasílaný objekt, tudíž jsem byl nucen změnit základní deserializátor za System.Xml.Serialization.XmlSerializer, který již nepožadoval jmenné prostory v hlavičce, tak jako základní deserializer. Po WebAPI přišlo na řadu testování Web Service, v tomto případě nabízí Visual Studio pohodlné napojení díky jazyku WSDL, pomocí kterého Visual Studio automaticky vygeneruje potřebné datové objekty. Zde při testování nenastal žádný problém. Výsledná časová náročnost implementace a testování je okolo 6 dní (do této časové náročnosti není započítána tvorba dokumentace).

4.3.4 Tvorba programátorské dokumentace

Posledním podúkolem bylo vytvoření uživatelské dokumentace. Při vytváření jsem spolupracoval s kolegou Ing. Michalem Prílepkem, PhD., který mi byl nápomocen při tvorbě struktury dokumentace. Dokumentace obsahuje popis jednotlivých objektů a jejich atributů, jak vstupních, tak i výstupních, dále výčty možných hodnot a dalších omezení hodnot atributů, v neposlední řadě také stručný popis napojení na jednotlivé webové aplikace. Dokumentace je dostupná online na webu společnosti VIKIPID[3].

4.4 Implementace eAPI platební brány ČSOB

Systém VIKIPID umožňuje zákazníkům e-shopů platbu za objednávku pomocí platební karty. Mým úkolem byla implementace propojení mezi systémem VIKIPID a eAPI ČSOB. ČSOB eAPI zprostředkovává přímé napojení na ČSOB platební bránu, která umožňuje platbu platební kartou, peněženkou MasterPass a platebními tlačítky ČSOB a Poštovní spořitelny. Zadáním bylo implementovat veškeré funkce eAPI nutné pro kompletní správu platebních transakcí, spolu s webovou aplikací, která bude zprostředkovávat toto propojení pro uživatele (nakupujícího). Další důležitou součástí bylo automatické generování a zasílání účtenek e-mailem zákazníkovi, spolu s potvrzením o úspěšném zaplacení objednávky. Povinnost vystavit účtenku vyplývá ze zákona o EET od 1. 3. 2017. Implementace napojení na eAPI a webové aplikace VIKIPID Brána zabrala přibližně 20 dní práce, kdy jedním z náročnějších úkolů bylo testování správnosti napojení, spolu se základním zabezpečením webové aplikace.

4.4.1 Implementace eAPI

Komunikace s eAPI probíhá přes HTTPS protokol, pomocí metod GET, POST a PUT, data jsou posílána ve formátu JSON a zabezpečena datovým podpisem, generovaným pomocí privátního RSA klíče. Komponenty pro práci s RSA klíči nabízí ČSOB přímo na svém webu. Samotný .NET Framework nabízí velkou škálu tříd a metod pro práci s RSA šifrováním ve jmenném prostoru System.Security.Cryptography. Nejtěžším úkolem byla implementace třídy, která bude zprostředkovávat univerzální generování podpisu zprávy. Z komponent .NET Frameworku jsem vytvořil třídu, která po vložení řetězce, který je sestaven z dat určených k zaslání zprávou, vytvoří podpis, zašifrovaný pomocí privátního klíče uloženého v souboru. Vytvořený podpis se poté zasílá spolu s daty. Na straně příjemce (eAPI) probíhá verifikace dat pomocí veřejného klíče (tento klíč tvoří dvojici s privátním na straně odesílatele zprávy). Stejně tak, při přijetí návratové zprávy z eAPI, je důležité verifikovat přijatou zprávu pomocí veřejného klíče platební brány. Tuto funkcionalitu jsem si plně odzkoušel pomocí testovací metody Echo, která slouží pouze pro zkoušku korektnosti napojení na eAPI, respektive platební bránu.

Samotná platební brána je provozována ve dvou verzích, jedna verze je určená pro testování napojení, druhá verze je produkční, určená pro ostrý provoz. Do produkčního prostředí lze přejít až poté, co jsou úspěšně provedeny specifikované operace s platbou v testovacím prostředí, čili v prvé řadě jsem měl možnost vygenerovat pouze testovací RSA klíče, se kterými jsem pracoval v testovacím prostředí, a až po řádném otestování a kontrole dat ze strany banky, jsem mohl vygenerovat produkční klíče k použití v produkční platební bráně.

Nejdůležitější operací eAPI je operace Init, jde o operaci, při které se zakládá platební transakce v platební bráně. Odpovědí na správně zaslaný požadavek, je návratový kód 0 a stav platby 1, čili založena. Operace Init má jako jediná speciální formát požadavku, tudíž bylo nutné vytvořit objekt pouze pro tento požadavek, ostatní operace mají jednotný formát zasílaného požadavku. Pokud je platební transakce úspěšně založena, je možné přejít k operaci Process, tato

operace probíhá pomocí přesměrování v prohlížeči zákazníka (metoda GET) na URL platební brány, kde následně proběhne platba. Po ukončení platby je zákazník přesměrován na URL adresu zadanou v požadavku operace Init, tudíž zpět na webovou aplikaci VIKIPID Brána. Do webové aplikace VIKIPID Brána je zaslána buď GET nebo POST návratová zpráva, o tom, která metoda bude použita, rozhoduje stav platby (zda proběhla korektně, byla zamítnuta aj.). Další důležitou operací je PaymentStatus, která vrací aktuální stav, ve kterém se platební operace nachází. Získání těchto stavů je potřebné pro kontrolu provedení platby a celkovou bezpečnost platebních operací v systému VIKIPID. V případě vrácení zásilky nebo jiné nastalé situace vyžadující vrácení platby zákazníkovi je nutné využít operaci eAPI Refund. Refund vrátí již zaúčtovanou platbu zpět na účet zákazníka. Tuto operaci lze navíc provést kdykoli - není časově omezena (například i měsíc po zaúčtování platby).

```
PUT https://iapi.iplatebnibrana.csob.cz/api/v1.7/payment/refund HTTP/1.1
Accept: application/json;charset=UTF-8
Content-Type: application/json
Host: iapi.iplatebnibrana.csob.cz
Content-Length: 436
Expect: 100-continue
{
  "merchantId":"M1MIPS0600",
  "payId":"6c6af6a324ea8CC",
  "dttm":"20170330204001",
  "signature":"J7ICFap699r3HJig9vduqPlyefhXP87oYxM1lonaLN1kK9VUnOGJwSo6cGT"
}
```

Výpis 4: Volání operace Refund eAPI pomocí HTTP metody PUT (podpis je zkrácený)

Od 1. 3. 2017 je zákonem daná povinnost evidovat platby, které byly zaplaceny platební kartou přes internet, tato povinnost se týkala i společnosti VIKIPID. Z tohoto důvodu bylo nutné implementovat rozšíření eAPI pro EET. Rozšíření pro EET se přidává formou atributu zasílaného požadavku, má samostatný kontrolní podpis, tudíž není součástí podpisu celého požadavku – je zcela nezávislé. Hlavní změna nastává při odesílání požadavku operace Init, kde je nutné vyplnit údaje plátce, nutné pro nahlášení tržby v systému Finanční správy ČR. Součástí odpovědi operace Init je FIK kód, popřípadě BKP kód, který je nutným údajem pro výdej účtenky zákazníkovi. Účtenka je zákazníkovi automaticky zasílána e-mailem pouze při úspěšném uhrazení objednávky a úspěšném nahlášení tržby. Další povinností je kontrola správnosti nahlášení údajů, to lze zjistit pomocí EET rozšíření odpovědi operace PaymentStatus, kde se nachází stav nahlášení konkrétní platby v systému Finanční správy. V případě, že dojde k vrácení platby pomocí operace Refund, je platba v systému Finanční správy automaticky odhlášena, a poté je znovu

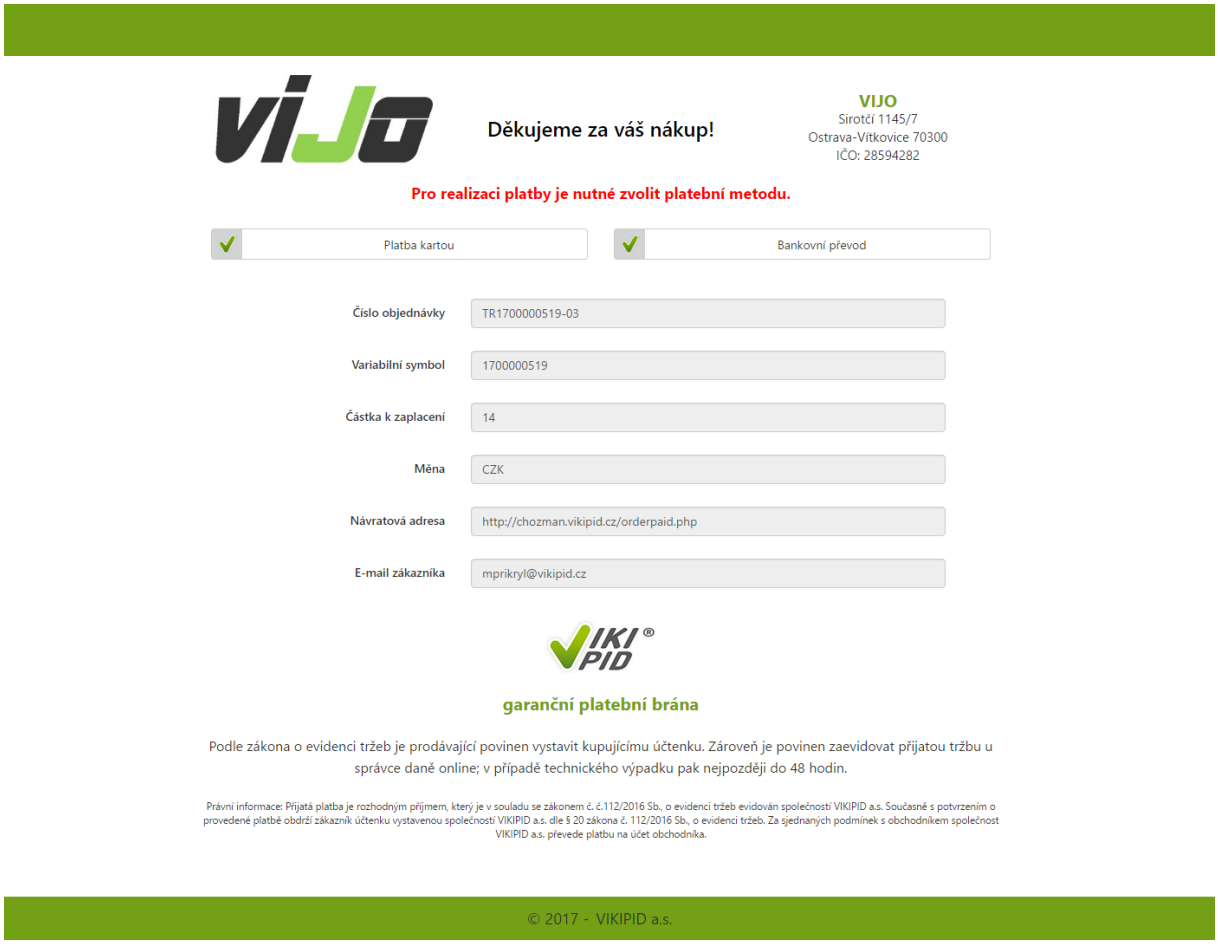
nutné pomocí operace PaymentStatus automaticky kontrolovat úspěšné odhlášení platby, kdy v případě neúspěchu odhlášení je okamžitě pomocí e-mailu informována IT podpora VIKIPID.

Samotné zpracování požadavků probíhá tak, že se objekt, který má stejnou strukturu, jakou má mít zasílaný požadavek, naplní daty a poté serializuje do řetězce ve formátu JSON. K serializaci je využita statická třída JsonConvert, ze jmenného prostoru Newtonsoft.Json, a metoda této třídy SerializeObject. Výsledný řetězec je odesílán pomocí metod POST nebo PUT a třídy .NET Frameworku System.Net.HttpWebRequest na rozhraní eAPI, ze kterého je poté přijata a zpracována odpověď. Pro požadavky odesílané metodou GET je využita třída .NET Frameworku System.Net.Http.HttpClient. Metoda GET s návratovou hodnotou je využita pouze u operace PaymentStatus, u všech ostatních operací je využito metod POST nebo PUT.

4.4.2 Webová aplikace VIKIPID Brána

Druhým podúkolem bylo vytvoření webové aplikace, která zprostředkuje zákazníkům platbu přes platební bránu ČSOB. Zadání bylo vytvořit jednoduché uživatelské rozhraní, s co nejmenším počtem kliknutí a s použitím firemních barev. Dalším důležitým aspektem bylo zobrazeni loga e-shopu, ze kterého byl zákazník přesměrován. Logo e-shopu je zobrazeno z důvodu, aby zákazník nabyl dojmu, že neopustil prostředí e-shopu a případně se nepolekal neznámé značky VIKIPID.

Při přesměrování zákazníka z e-shopu je zobrazena úvodní stránka (Obrázek 2), na které zákazník vidí logo a adresu e-shopu, u kterého nakoupil, základní informace o platbě a ve spodní straně právní informace ohledně evidování tržeb. Kliknutím na vybraný způsob platby je zákazník při výběru bankovního převodu přesměrován na informační stránku VIKIPID Brány, kde je zákazník informován o platebních údajích, nutných k platbě převodem. Zároveň s přesměrováním je zákazníkovi také odeslán e-mail s údaji nutnými k platbě. Po přečtení údajů, má zákazník možnost být přesměrován zpět na e-shop, který má možnost, dle návratové hodnoty umístěné v cílové URL adrese (adrese e-shopu), zobrazit stránku, jejíž obsah bude odpovídat významu návratové hodnoty. Při výběru platby kartou je na pozadí vyvolána operace eAPI Init, která založí platbu. Po úspěšném založení, je zákazník automaticky přesměrován na platební bránu ČSOB, kde může provést platbu za svou objednávku pomocí platební karty. Po ukončení samotné transakce, ať už úspěšném, či ne, je zákazník přesměrován zpět do VIKIPID Brány, kde je mu zobrazena stránka s dalšími instrukcemi. Při přesměrování je platební branou odeslána odpověď popisující konečný stav platební operace, tuto odpověď VIKIPID Brána zpracuje a zároveň se pro kontrolu znovu dotáže eAPI, pomocí operace PaymentStatus, na stav platební operace. V případě úspěšné platby je zákazníkovi zobrazena historie platby a stejně jako na stránce Bankovního převodu má možnost kliknutím na tlačítko být přesměrován zpět do e-shopu, taktéž s odpovídající návratovou hodnotou v URL adrese. V případě neúspěšné, či zamítnuté platby, je zákazník vyzván k opakované platbě, pomocí odkazu, který mu byl zaslán e-mailem při prvotním zpracování objednávky v systému VIKIPID, poté má také možnost být kliknutím na tlačítko přesměrován zpět do e-shopu.



Obrázek 2: Úvodní stránka VIKIPID Brány po přesměrování z e-shopu

Součástí webové aplikace VIKIPID Brána je také stránka se zkouškou konektivity, která vypíše, zda je webová aplikace správně napojena na eAPI a také, jestli je připojena k testovací nebo produkční verzi platební brány (eAPI). Tato kontrolní stránka slouží hlavně ke zpětné kontrole, zda byla na server nahrána správná verze, a aby nedošlo k nechtěným problémům s nesprávně vykonanou platební transakcí atd.

4.5 VIKIPID Intranet

Ve VIKIPID Intranetu lze kompletně spravovat systém VIKIPID, pod správou systému se skrývá obecně práce s daty, jako například vkládání informací pro správný a hladký běh systému. Nachází se zde i nejdůležitější agenda - agenda e-shopů, v této agendě se nachází veškeré informace o e-shopech využívajících systém VIKIPID. Mým úkolem bylo vytvořit tuto webovou aplikaci se vzhledem, podobným VIKIPID Účtu. Z důvodu, že byla předchozí verze aplikace VIKIPID Intranet vytvořena v jazyce PHP, byla součástí mého úkolu i implementace serverové části webové aplikace v jazyce C#, z tohoto důvodu si implementace vyžádala 25 dní práce.

4.5.1 Postup práce

Po konzultaci s kolegou jsme vytvořili harmonogram prací, podle kterého vývoj probíhal. Jako první měla vzniknout agenda e-shopů, jelikož je nejdůležitější a také nejsložitější. Další důležitou částí je Helpdesk, kde mohou uživatelé VIKIPID Účtu, prostřednictvím záložky Helpdesk, zanechat zprávu, kterou se musí zaměstnanci VIKIPIDu zabývat a případně odpovědět. Neméně důležitou agendou jsou Transakce. Tuto část Intranetu využívá převážně účetní, která zde čerpá informace a sestavy pro účetnictví. Poslední z důležitých částí jsou Reklamace. V reklamacích lze spravovat reklamace, které zadá zákazník při nespokojenosti s dodanými službami při online platbě bez dopravy. Zbylé části VIKIPID Intranetu není nutné přednostně detailně otestovat, z tohoto důvodu další harmonogram prací nebyl určen.

4.5.2 Popis VIKIPID Intranetu

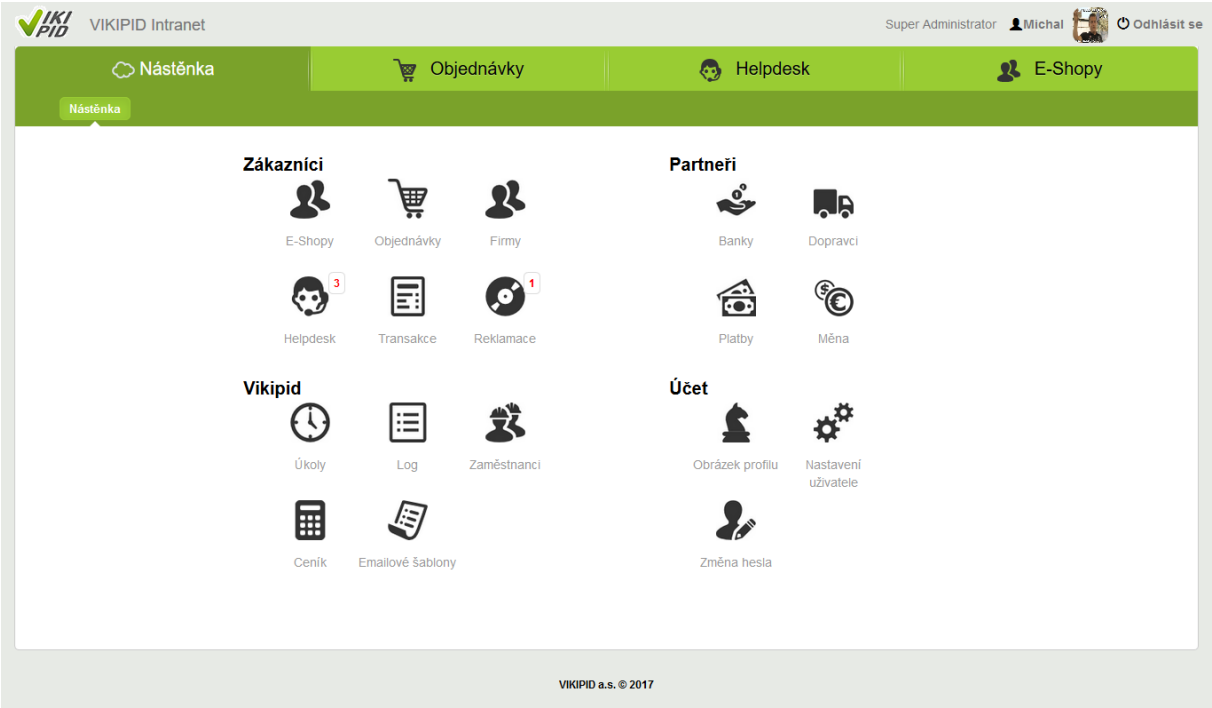
Aplikace je vyvíjena v ASP.NET MVC frameworku, napojení k databázi zajišťuje, tak jako v celém projektu, Entity framework. Nejprve krátce popíši princip MVC frameworku. Aplikace vytvořená v MVC frameworku je tvořena třemi hlavními komponentami, těmi jsou Model, View a Controller (z toho plyne i název MVC). MVC framework je založen na stejnojmenné softwarové architektuře, která odděluje uživatelské rozhraní, datový model a doménovou logiku do tří nezávislých komponent[4]. Model obsahuje datový model - logiku, výpočty validace, čili se jedná o klasickou C# třídu. View se stará o zobrazení výstupu uživateli, k zobrazení využívá jazyka HTML, View je v našem případě tvořen v Razor syntaxi, která je před zobrazením uživateli překládána na HTML kód. Ve view je taktéž možné využívat některé příkazy a konstrukce z jazyka C#, například cykly, podmínky, proměnné aj. Controller je prostředníkem mezi uživatelem, view a modelem, probíhá v něm zpracování dat, jak pro zobrazení k uživateli, tak zpracování od uživatele. Toto rozdělení kódu do 3 částí s sebou přináší lepší rozšiřitelnost a také větší přehlednost kódu.

Nyní se pokusím co nejvýstižněji popsat výše zmíněné nejdůležitější části VIKIPID Intranetu, ostatní části popíši pouze okrajově. Uživatelé jsou rozděleni do 4 rolí, jmenovitě super administrátor, administrátor, účetní a obchodní zástupce. Podle toho, v jaké roli je uživatel přihlášen, k takovému obsahu má přístup a má jej možnost i editovat. Tato autorizace je zajištěna pomocí spe-

ciální třídy, která dědí ze třídy .NET Frameworku System.Web.Mvc.AuthorizeAttribute, která zajišťuje metody pro autorizaci pomocí autorizačního atributu [CustomAuthorize] vloženého nad příslušnou metodu controlleru. Při vstupu do webové aplikace se nepřihlášenému uživateli se zobrazí přihlašovací stránka, kde zadá své přihlašovací údaje, ty se zašifrují a ověří se zašifrovanými údaji uloženými v databázi, pokud jsou údaje shodné, je uživatel přihlášen, zároveň je do jeho prohlížeče uložena autentizační cookie.

Po úspěšném přihlášení do aplikace je zaměstnanci zobrazena nástěnka (Obrázek 3), kde jsou zobrazeny aktivní ikony s názvy všech sekcí VIKIPID Intranetu, které může daný uživatel ve své uživatelské roli prohlížet a spravovat, kliknutím na ikonu je uživatel přesměrován do vybrané sekce. V prvním oddílu nástěnky, nazvaném Zákazníci, se nachází agenda e-shopů. Zde může zaměstnanec spravovat veškeré informace týkající se již dříve vložených e-shopů. První stránkou agendy e-shopů je seznam všech e-shopů, kde uživatel zvolí e-shop, který si přeje dále administrovat. Po úspěšném výběru e-shopu uvidí uživatel podrobné informace o e-shopu - jako kontaktní údaje, kontaktní osoby, obchodníkem vybrané přepravce, bankovní účty e-shopu aj. V podmenu se nachází seznam stránek spjatých s vybraným e-shopem. Kliknutím na určitý odkaz z podmenu má uživatel možnost přechodu do jiné sekce Intranetu s okamžitým zobrazením obsahu pouze pro daný e-shop (například objednávky, požadavky z helpdesku aj.). Součástí agendy e-shopů je také odesílání dokumentů, zde uživatel zvolí, které dokumenty si přeje odeslat nebo vygenerovat. Generuje se například smlouva o poskytování služeb, kde jsem využil knihoven určených pro práci s docx soubory přímo z programového prostředí, kdy vkládám data do předpřipraveného dokumentu a zadám do databáze příkaz pro odeslání tohoto dokumentu na vybraný e-mail. Agenda e-shopů je stále rozvíjena a vylepšována o nové funkce, jako poslední byla přidána možnost vkládání poznámek k e-shopu, tyto poznámky vidí pouze zaměstnanci firmy VIKIPID a to i historicky.

Další agendou z oddílu Zákazníci jsou Objednávky, zde si může uživatel zobrazit a dále filtrovat veškeré objednávky přijaté do systému VIKIPID. V sekci Firmy lze spravovat firemní údaje. V systému VIKIPID platí vztah, kdy jedna firma může mít více e-shopů, ale naopak e-shop může patřit pouze k jedné firmě čili firma vystupuje jako vlastník jednoho nebo více e-shopů. V sekci Helpdesk se nachází administrace požadavků, zadaných uživateli VIKIPID Účtu, na tyto požadavky lze odpovědět nebo je, po vyřešení, uzavřít. Pod každým požadavkem je možnost diskuze, kde mohou zaměstnanci VIKIPIDu komunikovat se zaměstnanci e-shopu (uživateli VIKIPID Účtu). Při přidání nového komentáře, bude vždy protistrana upozorněna ve svém webové aplikaci. V další sekci Transakce, se nachází informace o bankovních účtech systému VIKIPID. Jsou zde zobrazeny informace o pohybech na jednotlivých bankovních účtech získané ze systému Multicash Transfer (Kapitola 4.1). V sekci Reklamace se nachází zákaznické reklamace, zadané na základě nespokojenosti s dodanými službami e-shopu, který poskytuje služby VIKIPID. Reklamaci je možné zadat pouze při objednávce bez fyzické zásilky (bez dopravy), kdy není možné, nezávisle na e-shopu, zjistit úspěšné doručení zboží. Vytvořit reklamaci lze přes odkaz, doručený do e-mailu zákazníka při přijetí objednávky do systému VIKIPID. Po vložení reklamace má



Obrázek 3: Nástěnka VIKIPID Intranetu

zaměstnanec VIKIPIDu rozhodnout, zda je tato reklamace oprávněná, čímž vrátí peníze zpět zákazníkovi, nebo neoprávněná, čímž uvolní částku za objednávku k přeposlání e-shopu.

V oddílu Partneři se nachází sekce, kde lze spravovat dopravce, bankovní domy, možné platební měny a platby. V sekci banky má uživatel možnost editovat a přidávat údaje o bankovních ústavech, které jsou nutné pro Multicash transfer. V sekci Dopravci se nachází dopravci, jejichž služby nabízí systém VIKIPID. V sekci Měny se nachází měny, kterými lze hradit objednávky v systému VIKIPID. V poslední sekci s názvem platby se nachází zadané a zpracované platební transakce, které dále zpracuje nebo už zpracoval modul Multicash (Kapitola 4.1). Uživatel v roli super administrátora a administrátora má možnost v sekci platby manuálně vytvořit platbu na účet obchodníka a tím ji i připravit k odeslání.

Oddíl Vikipid slouží pro super administrátora a administrátora, kteří mají možnost zadávat úkoly jednotlivým zaměstnancům, nahlížet do chybových výpisů systému VIKIPID, přidávat a editovat zaměstnance, editovat ceník VIKIPID služeb a texty e-mailových šablon zasílaných, jak zaměstnancům, tak i zákazníkům.

Oddíl Účet slouží k administraci svého vlastního uživatelského účtu - nastavení profilového obrázku, nastavení informacích o uživateli a v neposlední řadě změnu hesla.

4.6 VIKIPID Účet

VIKIPID Účet je webová aplikace vytvořena v ASP.NET MVC frameworku, určená obchodníkům primárně ke správě objednávek, zaslaných ze svého e-shopu. Mým úkolem bylo vytvořit serverovou i klientskou část webové aplikace pro agendy svozových míst, uživatelů, smluvních dokumentů a hlavní nástěnku. Při vytváření zbylých částí webové aplikace jsem vypomáhal kolegovi. Součástí tohoto úkolu bylo také vytvoření uživatelské dokumentace. Tato dokumentace slouží k popisu jednotlivých částí uživatelského rozhraní, blíže popisuje jednotlivé stavy objednávky a popisuje postup vytvoření a zpracování objednávky ve VIKIPID Účtu. Čaová náročnost implementace webové aplikace spolu s uživatelskou dokumentací byla 14 dní, kdy 5 dní trvala implementace webové aplikace a zbylých 9 dní trvala tvorba dokumentace.

Uživatelé VIKIPID Účtu se dělí do rolí administrátor, skladník a účetní. Administrátor má plná přístupová práva do všech částí VIKIPID Účtu, skladník má práva ke správě objednávek a účetní má víceméně práva pouze k nahlížení do agendy objednávek.

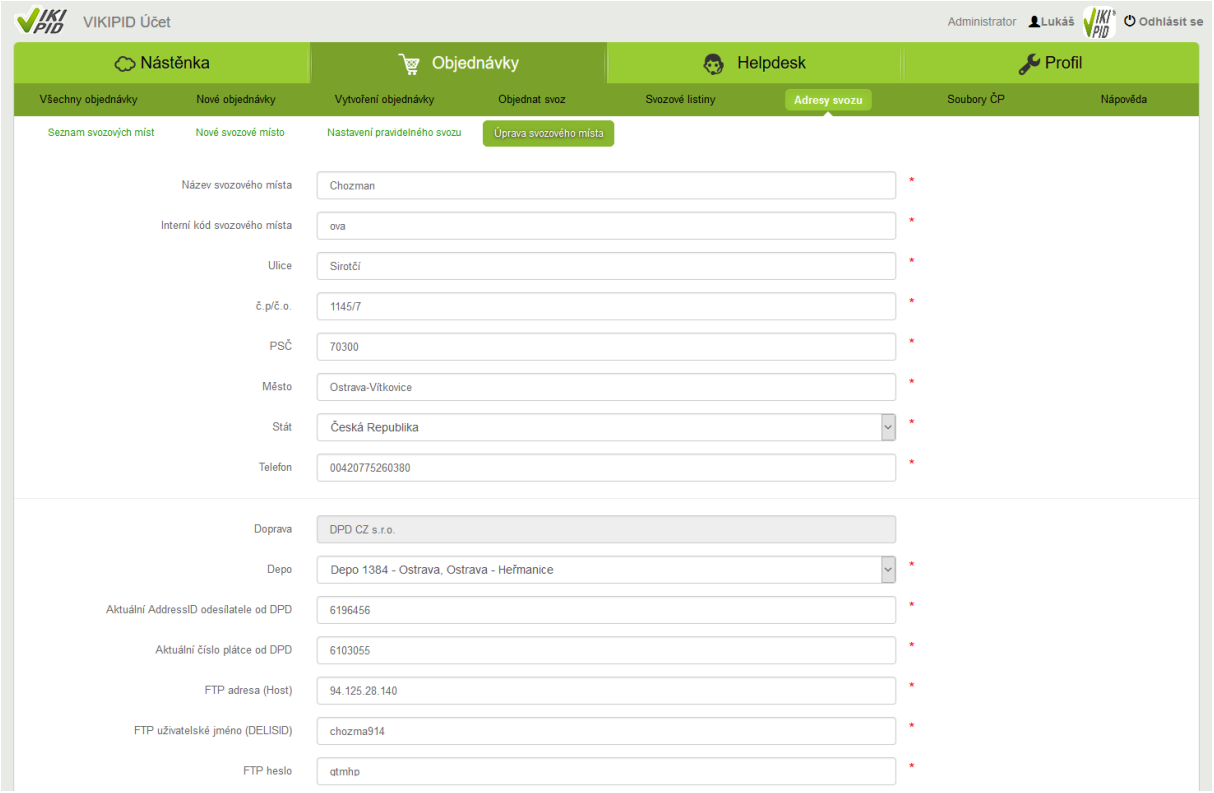
4.6.1 Popis VIKIPID Účtu

Pro přijetí objednávky do systému VIKIPID je nezbytné mít zadané a aktivní svozové místo. Svozová místa se zadávají ve webové aplikaci VIKIPID Účet, kde je pro ně zřízena samostatná agenda. Veškeré vkládání a nastavení svozových míst má možnost provádět pouze uživatel v roli administrátora, uživatelé v jiných rolích mohou do této sekce pouze nahlížet. Každé svozové místo má svou adresu, na této fyzické adrese posléze probíhá svoz zásilek, na které byl předem tento svoz objednán. Jedna svozová adresa může být přiřazena k více svozovým místům, ale pouze k tolika, s kolika dopravci má obchodník smlouvu (momentální maximum svozových míst je 5). K tomu, aby bylo možné odeslat samotnou zásilku i žádost o svoz, je nutné vyplnit atributy svozového místa, různé pro každého dopravce. Jedná se o atributy jako přihlašovací údaje k webovým službám, unikátní URL pro Track & Trace aj. Jak jsem již dříve zmínil, bez těchto údajů není možné odeslat data o zásilce, a tím pádem ani vyskladnit a odeslat objednávku.

Při přidávání svozového místa je možné vybrat z již existujících svozových adres nebo založit adresu novou. K vybrané svozové adrese se poté zobrazí dopravci, pro něž lze založit svozové místo (pouze ti dopravci, pro které prozatím není svozové místo k této adrese založeno). Po výběru dopravce se uživateli zobrazí textové pole pro vyplnění jednotlivých atributů. Z těchto důvodů je nutné při zadávání nového svozového místa kontrolovat správnost i samotné vyplnění povinných údajů pro vybraného dopravce. Kontrola správnosti a vyplnění povinných údajů dopravce probíhá až po odeslání formuláře uživatelem na server. Kontrola vyplnění svozové adresy a správnosti vložených údajů probíhá dynamicky již při zadávání, pomocí jQuery unobtrusive validation. V případě chyby je uživateli pod textovým polem obsahujícím chybu zobrazena odpovídající chybová zpráva. Chybová zpráva je zadaná v modelu (C# třídě), spolu se specifikací zobrazeného atributu. Pro případ, že má uživatel například v prohlížeči vypnutý javascript, probíhá kontrola vložených údajů znovu na serveru, kdy v případě nekorektních údajů není svozové

místo uloženo a uživateli je zobrazena chybová zpráva. Obdobně probíhá i editace svozového místa (Obrázek 4), kde je validace vstupu uživatele prováděna stejně jako u vytváření svozového místa.

Důležitou součástí svozových míst je nastavení pravidelného svozu. Zde si uživatel zvolí svozovou adresu, pro kterou si přeje nastavit pravidelný svoz. Na nové stránce je uživateli zobrazeno nastavení pro všechny dopravce (pro každé svozové místo) svázané se svozovou adresou. Zde si zakliknutím vybraných dní v týdnu zvolí, na které dny mu bude systém VIKIPID automaticky objednávat svoz, do kterého bude moci přidávat zásilky. Automatické objednávání svozu je zajištěno servisem, automaticky spouštěném několikrát denně ve stanovený čas.



Obrázek 4: Úprava svozového místa ve VIKIPID Účtu

V agendě uživatelé má uživatel v roli administrátora možnost spravovat uživatele, jejich vyplněné údaje nebo jim zakázat přístup k VIKIPID Účtu. Ostatní uživatelské role mohou měnit pouze své vlastní údaje. Všechny uživatelské role mají možnost si změnit své údaje jako - titul, jméno, příjmení, e-mail a firemní funkci. Pouze uživatel v roli administrátora má možnost změnit roli konkrétního uživatele. Uživatel v roli administrátora má také možnost vytvořit nový uživatelský účet. Při vytváření účtu je možné zvolit uživatelské jméno, které musí být unikátní v rámci celého systému VIKIPID, kdy toto jméno nelze v budoucnu změnit. Každý uživatel má v této sekci dále možnost změny hesla, vytvoření nového hesla (na základě potvrzení platného e-mailu) a změnu svého profilového obrázku.

V agendě smluvních dokumentů je na vstupní stránce zobrazen přehled dokumentů, které se týkají obchodního vztahu mezi obchodníkem a firmou VIKIPID a.s. Obchodník má možnost zde nalézt aktuální obchodní podmínky, ceník VIKIPID služeb a všechny smlouvy a dodatky uzavřené mezi vlastníkem e-shopu a firmou VIKIPID. Všechny dokumenty jsou uloženy na serveru ve formátu PDF, k jejichž zobrazení je využit HTML tag object (metoda pro načítání PDF, Výpis 5), který umožňuje vložit a zobrazit na HTML stránce multimédia jako zvuky, videa, PDF a mnoho dalších. Jediný dokument generovaný při zobrazení je ceník VIKIPID služeb, který je zobrazován dynamicky pomocí HTML, dle aktuálních ceníkových hodnot z databáze a aktuální ceníkové skupiny obchodníka.

```
[CustomAuthorize(Roles = "SettingsController")]
public ActionResult GetContractPDF(string file)
{
   try
   {
      FileStream fileStream;
      if (System.IO.File.Exists(file))
      {
        fileStream = new FileStream(file, FileMode.Open, FileAccess.Read);
      }
      else
      {
       fileStream = new FileStream("C:\\d.pdf",FileMode.Open, FileAccess.Read);
      }
      return File(fileStream, "application/pdf");
    }
    catch (Exception e)
    {
      CreateVikipidException(e, "GetContractPDF",
      $"Nepodarilo se zobrazit smlouvu, user ID - {User.UserId}.", "Document");
      AddMessageToTempData("Pri nacteni smlouvy nastala chyba.", true);
      return RedirectToAction("Index", "Document");
    }
}
```

Výpis 5: Metoda controlleru s načtením PDF souboru pro jeho zobrazení na HTML stránce

Poslední částí, kterou jsem vytvářel, byla hlavní nástěnka. Hlavní nástěnka je skupina stránek, která se uživateli zobrazí po úspěšném přihlášení do VIKIPID Účtu. Jako první se uživateli po přihlášení zobrazí stránka s aktualitami, v aktualitách je zákazník informován o novinkách v

systému VIKIPID nebo různých dalších informacích. Nepřečtené aktuality jsou zobrazeny červeně a přečtené aktuality jsou zobrazeny černě. Jednotlivé aktuality má uživatel možnost si také zobrazit v detailu, čímž i potvrdí přečtení aktuality. Jako druhá se nachází v podmenu stránka Upozornění, na této stránce bude uživatel upozorněn v případě, že se některá číselná řada dopravce (momentálně pouze PPL) blíží ke svému konci nebo již nenabízí žádné další volná čísla k přiřazení zásilkám. Pokud si obchodník nevyžádá u dopravce novou číselnou řadu a nenahlásí ji svému obchodnímu zástupci, nebude možné odeslat data o zásilce do systému dopravce. Další stránkou podmenu je Přehled. V Přehledu nalezne obchodník historické součty zásilek, dle rozličných kategorií (například dle platby, dle stavu objednávky atd.) vztahující se vždy k aktuálnímu dni. V další záložce s názvem Faktury, má obchodník možnost zobrazení faktur vygenerovaných fakturačním softwarem. Na stránce Faktury lze také získat informace o uhrazení faktury další informace o jednotlivých fakturách. Poslední záložkou jsou Služby, kde se nachází dvě tabulky. V první tabulce najdeme výpis VIKIPID služeb, spolu s informací, zda je daná služba momentálně aktivní, či neaktivní. V druhé tabulce se nachází výpis dopravců, které má e-shop vybrány v rámci systému VIKIPID. V tabulce dopravců se taktéž nachází informace, zda je možné služby využívat. V případě, že e-shop jakkoli poruší nutné podmínky pro poskytování služeb VIKIPID, dozví se o omezení poskytovaných služeb na této stránce, případně v aktualitách.

5 Dovednosti a znalosti uplatněné v průběhu odborné praxe

Při vykonávání odborné praxe jsem využil mnoho znalostí získaných při studiu na univerzitě. Velká část znalostí pocházela z předmětů Programování II, Programovací jazyky I a Programovací jazyky II, v těchto předmětech jsem získal znalosti z oblasti objektového programování, které jsem na praxi zužitkoval. Při mé praxi jsem převážně využíval programovací jazyk C# společně s .NET frameworkem, kdy většina mých znalostí o tomto jazyce a technologiích pocházela z předmětů Programovací jazyky II, kde jsem si osvojil hlavně základy programování v jazyce C#, a také ze předmětu Architektura technologie .NET, kde jsem získal pokročilejší znalosti o technologiích .NET Frameworku. Při třídním návrhu jsem využil znalosti návrhových vzorů získané z předmětu Vývoj informačních systémů. Znalosti získané v předmětech Úvod do databázových systémů a Databázové a informační systémy jsem využil při návrhu struktur databáze a také při tvorbě SQL dotazů. Také při vypracování této práce jsem využil znalosti nabyté na univerzitě, konkrétně v předmětu Elektronické publikování, kde jsem získal znalosti LaTeXu.

6 Dovednosti a znalosti scházející v průběhu odborné praxe

U některých zadaných úkolů bylo nutné spolupracovat v týmu, zde mi scházela zkušenost spolupráce s více programátory, jelikož jsem doposud na školních projektech pracoval převážně sám. Nejvíce znalostí jsem postrádal v oblasti webových technologií. V rámci vývoje jsem také vytvářel webové stránky v jazyce HTML, potažmo pomocí ASP.NET Razor syntaxe. Mé znalosti jazyka HTML byly při začátku praxe pouze základní, taktéž znalosti kaskádových stylů. Neznalost jazyka HTML a CSS jsem však mohl vyřešit volbou vhodných předmětů. Dalším nedostatkem byla počáteční neznalost MVC frameworku a RESTFul API, tyto neznalosti jsem se snažil eliminovat studováním poměrně rozsáhlé dokumentace a tvorbou vlastních aplikací.

Při odborné praxi jsem se také poprvé setkal s možností sdílení a verzování kódu pomocí cloudových služeb, konkrétně jsem se setkal s Microsoft TFS. Tyto služby mají velice kladný přínos pro práci v týmu.

7 Závěr

Při nástupu do společnosti VIKIPID a.s. jsem neoplýval mnoho praktickými znalostmi a zkušenostmi v oblasti programování a práce v týmu. Právě absolvováním oné odborné praxe jsem získal potřebnou praxi, která je dnes velmi ceněná na trhu práce. Naučil jsem se používat webové technologie, prohloubil jsem znalosti a zkušenosti programování v jazyce C#, okusil jsem práci v týmu a setkal se s moderními technologiemi pro vývoj v týmu.

Ze začátku jsem byl nucen dohánět scházející znalosti samostudiem, ale postupně jsem tyto neznalosti eliminoval, i když potřeba samostudia je stále nutná, jelikož v informatice se lze neustále zdokonalovat. Konzultant Bc. Michal Sloviak mi svěřil i důležité úkoly, jako například implementace VIKIPID Brány (Kapitola 4.4) nebo VIKIPID Účtu (Kapitola 4.6), kdy mi byla dána velká odpovědnost a důvěra. Veškeré aplikace a úkoly, na kterých jsem se podílel, jsem zdárně dokončil a předal jako plně funkční k reálnému produkčnímu užití.

Při praxi jsem se nesetkal pouze s programováním, ale i se základní správou serverového operačního systému Windows Server 2012, správou SŘBD Microsoft SQL Server nebo s reálným nasazením webové aplikace na webový server.

Celkově jsem byl s výběrem tématu mé bakalářské práce i s průběhem praxe velmi spokojen.

Literatura

[1] O společnosti VIKIPID a.s.
VIKIPID [online]. [cit. 2016-03-17].
http://www.vikipid.cz

[2] XPath úvod
w3schools [online]. [cit. 2016-03-23].
https://www.w3schools.com/xml/xpath_intro.asp

[3] VIKIPID technická dokumentace
VIKIPID technická dokumentace - datová komunikace [online]. [cit. 2016-03-27].
http://www.vikipid.cz/documents/technicka_dokumentace.pdf

[4] MVC úvod
ITnetwork [online]. [cit. 2016-04-08].
http://www.itnetwork.cz/csharp/asp-net/mvc/asp-dot-net-uvod-do-mvc-architektury